

# Chương 12

## KẾ THỪA

## **0. MỤC TIÊU**

- Hiểu được các loại quan hệ?
- Hiểu được kế thừa trong lập trình hướng đối tượng là gì?
- Hiểu được khái niệm cây kế thừa.
- Hiểu được khái niệm sơ đồ lớp.

# 1. QUAN HỆ

Người ta chia các quan hệ thành những loại như sau:

- Quan hệ một một (1-1)
- Quan hệ một nhiều (1-n)
- Quan hệ nhiều nhiều (m-n)
- Quan hệ đặt biệt hóa, tổng quát hóa.

## 1.1. QUAN HỆ MỘT MỘT (1-1)

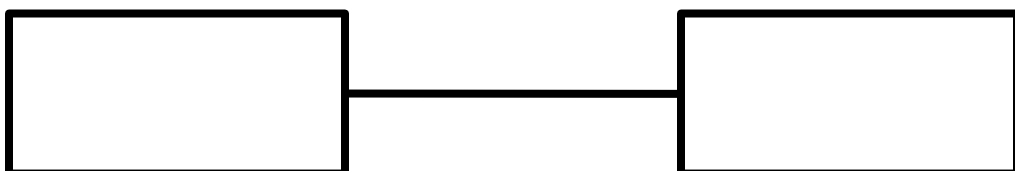
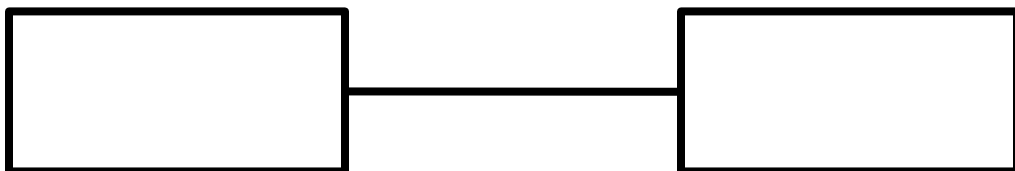
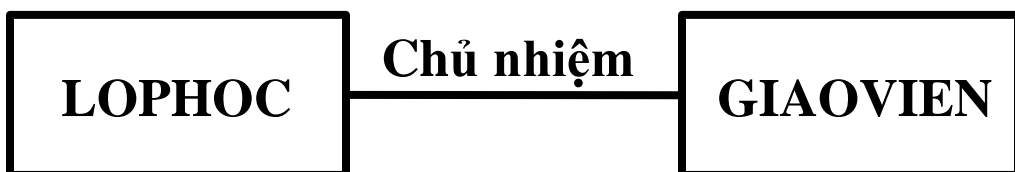
- **Khái niệm:** Hai lớp đối tượng được gọi là quan hệ một-một với nhau khi một đối tượng thuộc lớp này quan hệ với một đối tượng thuộc lớp kia và một đối tượng thuộc lớp kia quan hệ duy nhất với một đối tượng thuộc lớp này.
- Hình vẽ



- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với một đối tượng thuộc lớp B và một đối tượng lớp B quan hệ duy nhất với một đối tượng thuộc lớp A.

## 1.1 QUAN HỆ MỘT MỘT (1-1)

– Ví dụ minh họa



## 1.2. QUAN HỆ MỘT NHIỀU (1-n)

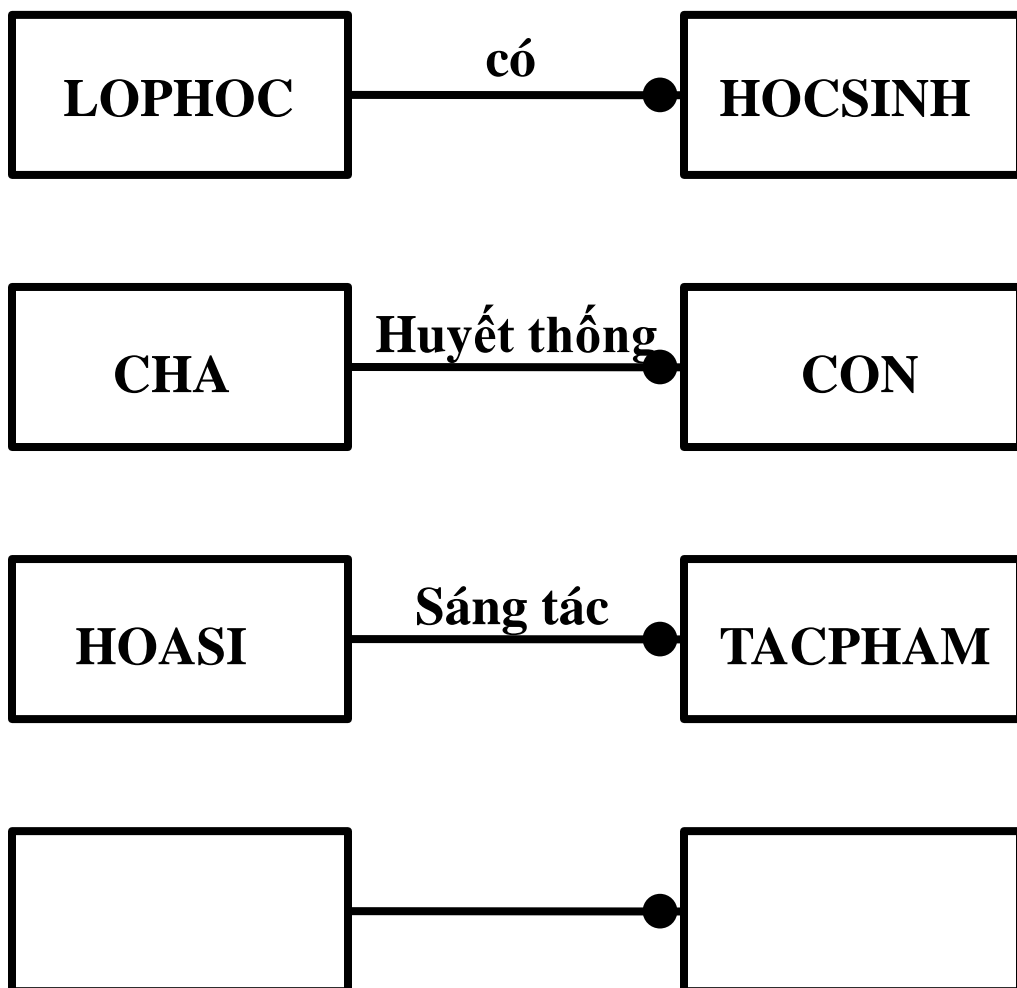
- **Khái niệm:** Hai lớp đối tượng được gọi là quan hệ một-nhiều với nhau khi một đối tượng thuộc lớp này quan hệ với nhiều đối tượng thuộc lớp kia và một đối tượng lớp kia quan hệ duy nhất với một đối tượng thuộc lớp này.
- Hình vẽ



- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với nhiều đối tượng thuộc lớp B và một đối tượng lớp B quan hệ duy nhất với một đối tượng thuộc lớp A.

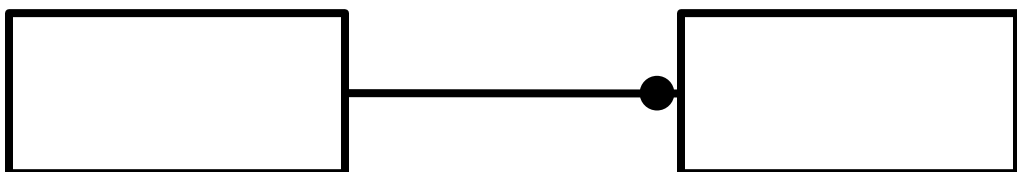
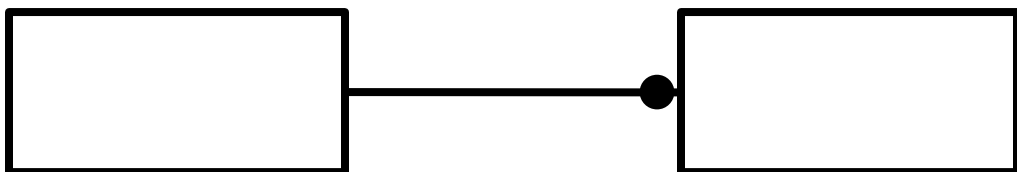
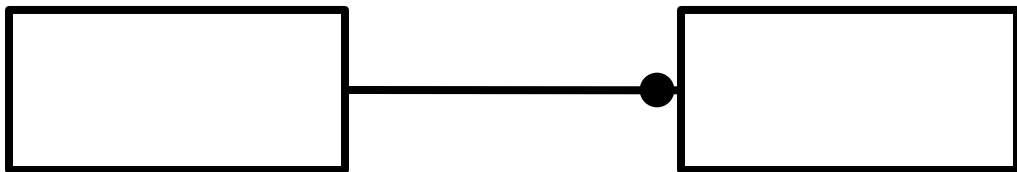
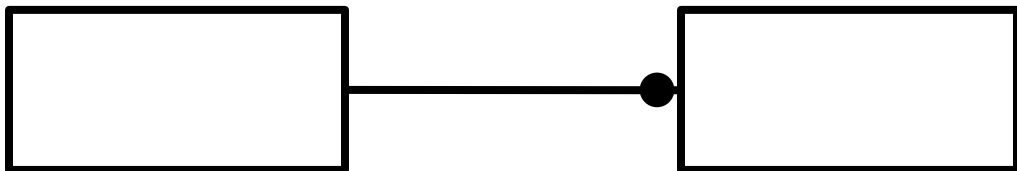
## 1.2 QUAN HỆ MỘT NHIỀU (1-n)

– Ví dụ minh họa:



## 1.2 QUAN HỆ MỘT NHIỀU (1-n)

– Ví dụ minh họa:





## 1.3 QUAN HỆ NHIỀU NHIỀU (m-n)

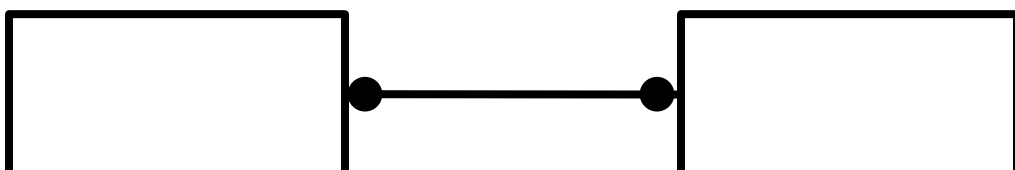
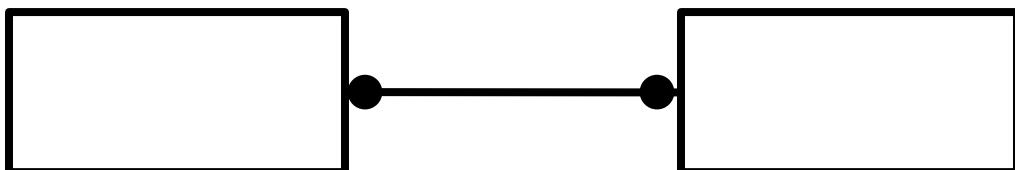
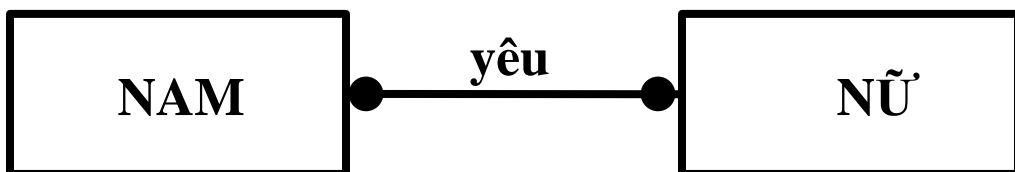
- **Khái niệm:** hai lớp đối tượng được gọi là quan hệ nhiều-nhiều với nhau khi một đối tượng thuộc lớp này quan hệ với nhiều đối tượng thuộc lớp kia và một đối tượng lớp kia cũng có quan hệ với nhiều đối tượng thuộc lớp này.
- Hình vẽ



- Trong hình vẽ trên ta nói: một đối tượng thuộc lớp A quan hệ với nhiều đối tượng thuộc lớp B và một đối tượng lớp B cũng có quan hệ với nhiều đối tượng thuộc lớp A.

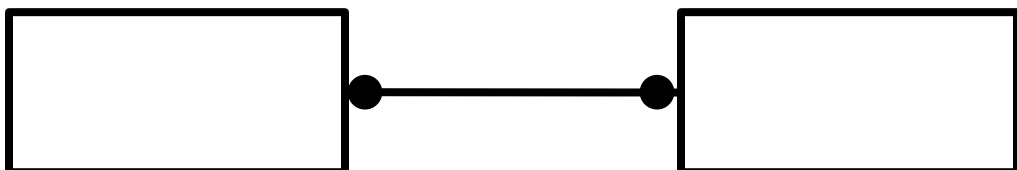
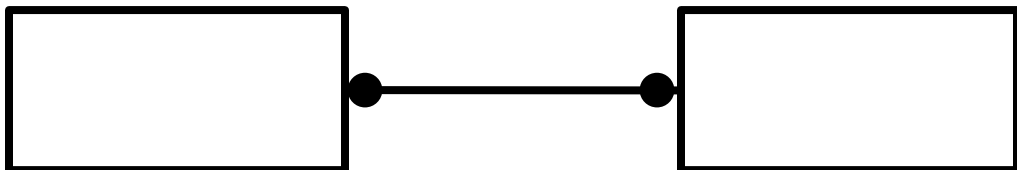
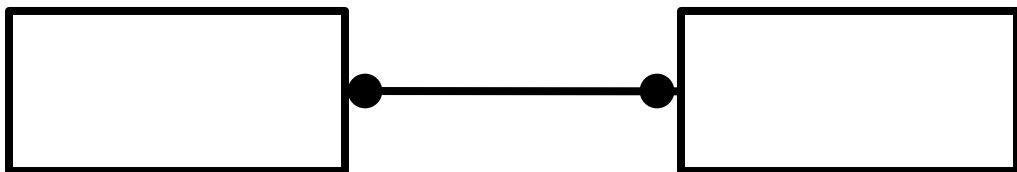
## 1.3 QUAN HỆ NHIỀU NHIỀU (m-n)

– Ví dụ minh họa:



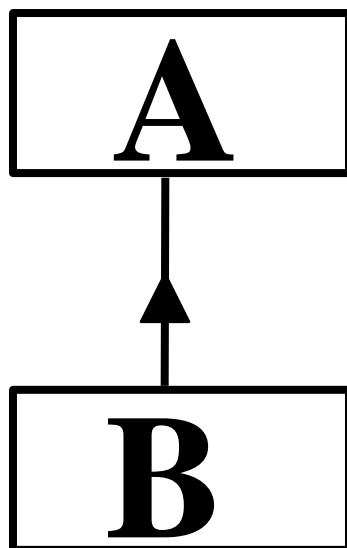
## 1.3 QUAN HỆ NHIỀU NHIỀU (m-n)

– Ví dụ minh họa:



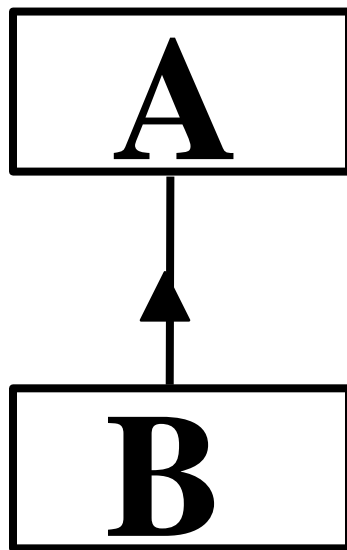
## 1.4. QUAN HỆ ĐẶC BIỆT HÓA-TỔNG QUÁT HOÁ

- **Khái niệm:** hai lớp đối tượng được gọi là quan hệ đặc biệt hóa-tổng quát hóa với nhau khi, lớp đối tượng này là trường hợp đặc biệt của lớp đối tượng kia và lớp đối tượng kia là trường hợp tổng quát của lớp đối tượng này.
- Hình vẽ



## 1.4. QUAN HỆ ĐẶT BIỆT HÓA-TỔNG QUÁT HOÁ

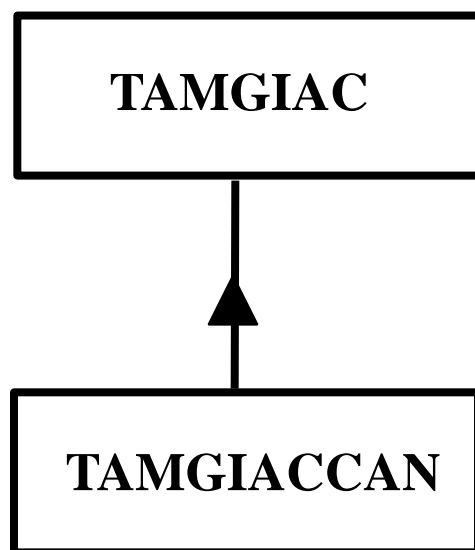
- Hình vẽ



- Trong hình vẽ trên ta nói: lớp đối tượng B là trường hợp đặc biệt của lớp đối tượng A và lớp đối tượng A là trường hợp tổng quát của lớp đối tượng B.

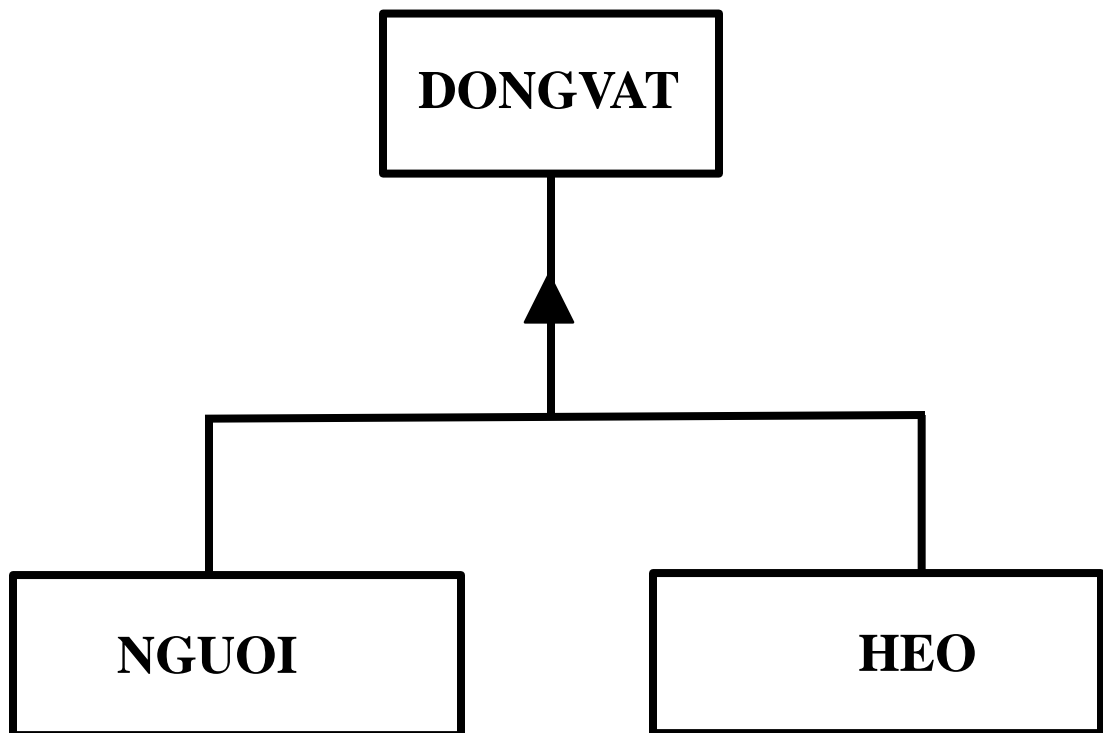
## 1.4. QUAN HỆ ĐẶT BIỆT HÓA-TỔNG QUÁT HOÁ

– Ví dụ 1:



## 1.4. QUAN HỆ ĐẶT BIỆT HÓA-TỔNG QUÁT HOÁ

– Ví dụ 2:

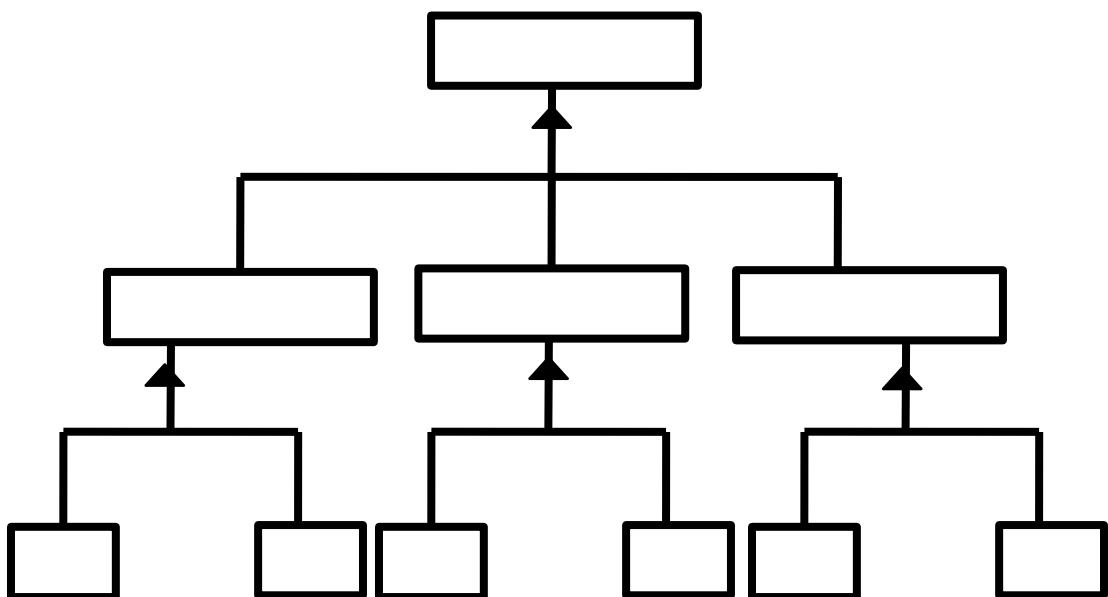


## 2. CÂY KẾ THỪA

- **Khái niệm:** Cây kế thừa là một cây đa nhánh thể hiện mối quan hệ đặc biệt hóa-tổng quát hóa giữa các lớp trong hệ thống, chương trình.
- Ví dụ: Hãy vẽ cây kế thừa cho các lớp đối tượng sau:
  - + Lớp XEDAP                      • Lớp XELAM
  - + Lớp XEGANMAY               • Lớp XE
  - + Lớp XEHOI                     • Lớp XEBABANH
  - + Lớp XEHAIBANH              • Lớp XEBONBANH
  - + Lớp XETAINHE                • Lớp XEXICHLO



## 2. CÂY KẾ THỪA (tiếp)



### 3. SƠ ĐỒ LỚP

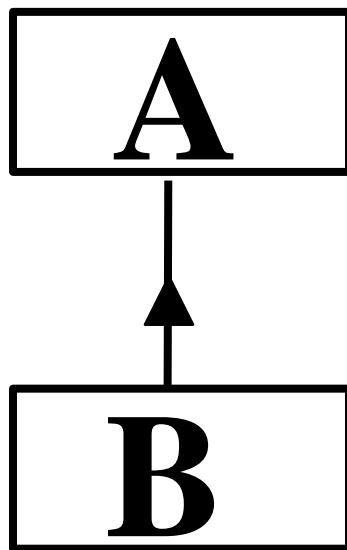
- **Khái niệm:** Sơ đồ lớp là sơ đồ thể hiện tất cả các mối quan hệ giữa các lớp trong hệ thống, chương trình.
- Ví dụ minh họa: Hãy vẽ sơ đồ lớp cho các lớp đối tượng sau:
  - + Lớp GIAOVIEN
  - + Lớp HOCSINH
  - + Lớp LOPHOC
  - + Lớp MONHOC
  - + Lớp NHANVIEN: tất cả những người làm việc trong trường.
  - + Lớp CNV: là những người làm việc trong nhà trường nhưng ko trực tiếp đứng lớp. Ví dụ: Bảo vệ, lao công, bảo mẫu, ...

## **3. SƠ ĐỒ LỚP**

## 4. KẾ THỪA TRONG C++

- Thế giới thực
- Lập trình hướng đối tượng với C++
- Phạm vi truy xuất
- Từ khoá dẫn xuất

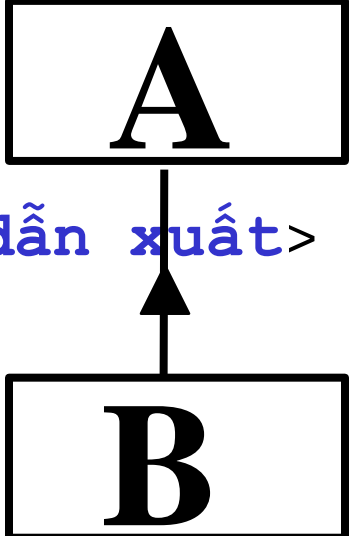
## 4.1 THỂ GIỚI THỰC



- Trong hình vẽ trên ta nói A và B có quan hệ đặc biệt hoá, tổng quát hoá với nhau. Trong đó B là trường hợp đặt biệt của A, và A là trường hợp tổng quát của B.

## 4.2 LTHĐT VỚI C++

```
1. class A
2. {
3.     ...
4. };
5. class B:<từ khóa dẫn xuất> A
6. {
7.     ...
8. };
```



```
classDiagram
    A --|> B
```

- Trong khai báo trên ta nói lớp B kế thừa từ lớp A.
- Lớp đối tượng A được gọi là lớp cơ sở.
- Lớp đối tượng B được gọi là lớp dẫn xuất từ lớp đối tượng A.

## 4.3 PHẠM VI TRUY XUẤT

- Một thuộc tính hay một phương thức khi được khai báo trong một lớp ta có thể khai báo trong 3 phạm vi khác nhau: **private**, **public** hoặc **protected**.
- Về mặt nguyên tắc cho tới thời điểm này thì một thuộc tính hay một phương thức khi được khai báo trong phạm vi **private** hay **protected** thì tương đương nhau. Nghĩa là, thuộc tính và phương thức được khai báo trong hai phạm vi này chỉ được phép truy xuất bên trong lớp mà thôi và không được quyền truy xuất từ bên ngoài lớp.

## 4.3 PHẠM VI TRUY XUẤT

- Ví dụ: Hãy cho biết trong đoạn chương trình sau câu lệnh nào đúng, câu lệnh nào sai.

```
1. class A
2. {
3.     private:
4.         int a;
5.         void f();
6.     protected:
7.         int b;
8.         void g();
9.     public:
10.        int c;
11.        void h();
12. } ;
```



## 4.3 PHẠM VI TRUY XUẤT

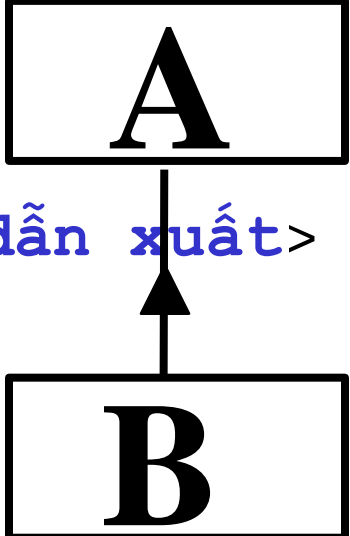
```
13. void A::f( )
14. {
15.     a = 1;
16.     b = 2;
17.     c = 3;
18. }
19. void A::g( )
20. {
21.     a = 4;
22.     b = 5;
23.     c = 6;
24. }
25. void A::h( )
26. {
27.     a = 7;
28.     b = 8;
29.     c = 9;
30. }
```

## 4.3 PHẠM VI TRUY XUẤT

```
31. void main( )
32. {
33.     A x;
34.     x.a = 10;
35.     x.f( ) ;
36.     x.b = 20;
37.     x.g( ) ;
38.     x.c = 30;
39.     x.h( ) ;
40. }
```

## 4.4 TỪ KHÓA DẪN XUẤT

```
1. class A
2. {
3.     ...
4. };
5. class B:<từ khóa dẫn xuất> A
6. {
7.     ...
8. };
```



- Trong khai báo trên ta nói lớp B kế thừa từ lớp A.
- Lớp đối tượng A được gọi là lớp cơ sở.
- Lớp đối tượng B được gọi là lớp dẫn xuất từ lớp đối tượng A.

## 4.4 TỪ KHÓA DẪN XUẤT

- Trong ngôn ngữ C++ có ba loại *từ khóa dẫn xuất* đó là: **private**, **protected** và **public**. Thông thường trong thực tế người ta hay sử dụng từ khóa dẫn xuất **public** là nhiều nhất.

## 4.4 TỪ KHÓA DẪN XUẤT

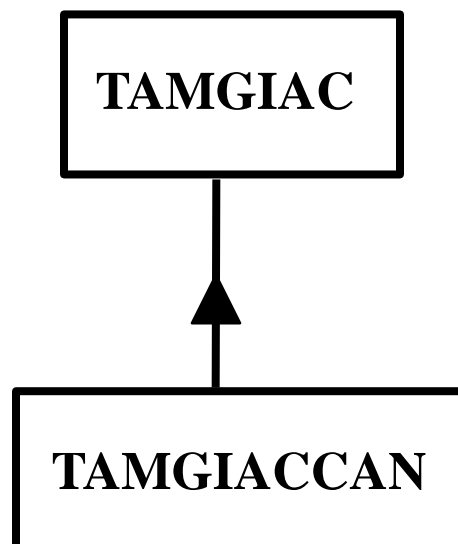
- Ví dụ 01: Khai báo lớp tam giác và lớp tam giác cân.

1. **class CTamGiac**

```
2. {  
3. |    ...  
4. } ;
```

5. **class CTamGiacCan:public**  
**CTamGiac**

```
6. {  
7. |    ...  
8. } ;
```



## 4.4 TỪ KHÓA DẪN XUẤT

- Ví dụ 02: Khai báo lớp động vật, lớp heo và lớp người.

11. **class CDongVat**

```
12. {  
13. |    ...  
14. } ;
```

15. **class CHeo:private CDongVat**

```
16. {  
17. |    ...  
18. } ;
```

19. **class CNgnoi:public CDongVat**

```
20. {  
21. |    ...  
22. } ;
```

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

F C			

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Phạm vi lớp cơ sở		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.



## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Phạm vi lớp cơ sở		
Private		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Phạm vi lớp cơ sở		
Private		
Protected		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Phạm vi lớp cơ sở		
Private		
Protected		
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất		
Phạm vi lớp cơ sở		
Private		
Protected		
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khóa dẫn xuất	Private	
Phạm vi lớp cơ sở		
Private		
Protected		
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Phạm vi lớp cơ sở	Từ khoá dẫn xuất	Private	Public
	Private		
	Protected		
	Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất	Private	Public
Phạm vi lớp cơ sở		
Private		
Protected		
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất	Private	Public
Phạm vi lớp cơ sở		
Private		
Protected	private	
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.



## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất	Private	Public
Phạm vi lớp cơ sở		
Private		
Protected	<b>private</b>	<b>protected</b>
Public		

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất	Private	Public
Phạm vi lớp cơ sở		
Private		
Protected	<b>private</b>	<b>protected</b>
Public	<b>private</b>	

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

Từ khoá dẫn xuất	Private	Public
Phạm vi lớp cơ sở		
Private		
Protected	<b>private</b>	<b>protected</b>
Public	<b>private</b>	<b>public</b>

- Ghi chú: Từ khoá dẫn xuất có ba loại là private, protected, public. Các sinh viên tự tìm hiểu thêm từ khoá dẫn xuất protected trong tài liệu.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Các thuộc tính và phương thức được khai báo trong phạm vi *private* của lớp cơ sở thì sẽ không được hiểu ở lớp dẫn xuất.
- Các thuộc tính và phương thức được khai báo trong phạm vi *protected* của lớp cơ sở nếu được dẫn xuất bằng từ khóa *private* thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần *private* của lớp dẫn xuất.
- Các thuộc tính và phương thức được khai báo trong phạm vi *protected* của lớp cơ sở nếu được dẫn xuất bằng từ khóa *public* thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần *protected* của lớp dẫn xuất.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Các thuộc tính và phương thức được khai báo trong phạm vi *public* của lớp cơ sở nếu được dẫn xuất bằng từ khóa *private* thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần *private* của lớp dẫn xuất.
- Các thuộc tính và phương thức được khai báo trong phạm vi *public* của lớp cơ sở nếu được dẫn xuất bằng từ khóa *public* thì các thuộc tính và phương thức đó sẽ được hiểu ở lớp dẫn xuất như là thành phần *public* của lớp dẫn xuất.

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

<div>Từ khóa dẫn xuất</div> <div>Phạm vi lớp cơ sở</div>	Private	Public
Private	(1)	(2)
Protected	(3)	(4)
Public	(5)	(6)

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

	(1)	(2)
	(3)	(4)
	(5)	(6)

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++



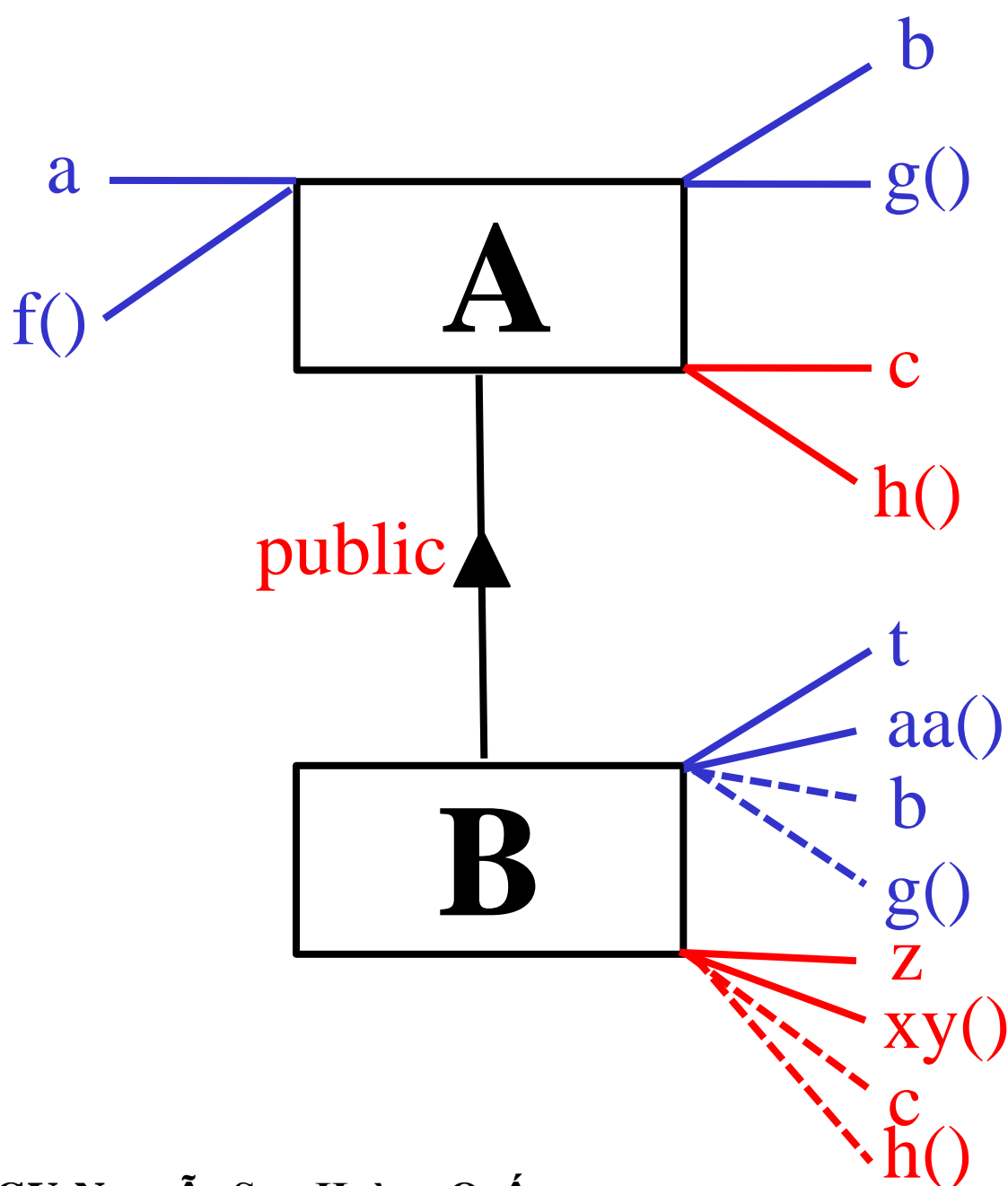

## 5. QUI TẮC KẾ THỪA TRONG C++

- Bảng qui tắc kế thừa trong C++

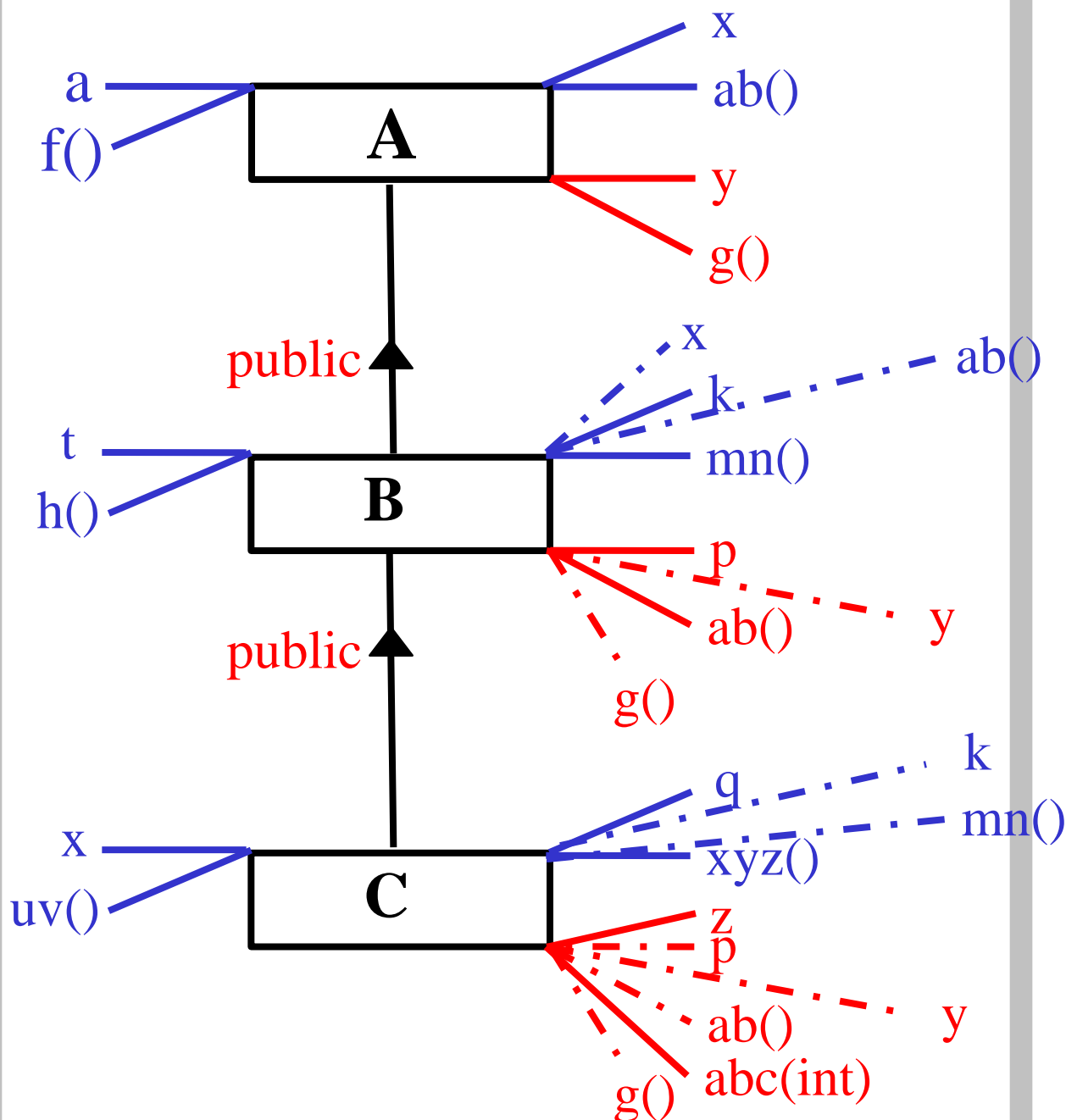
## 6. CÂY KẾ THỪA CHI TIẾT

- Qui tắc vẽ cây kế thừa chi tiết:
  - + Các thuộc tính và phương thức thuộc phạm vi **private** được vẽ với **màu xanh** bên trái.
  - + Các thuộc tính và phương thức thuộc phạm vi **protected** được vẽ với **màu xanh** bên phải.
  - + Các thuộc tính và phương thức thuộc phạm vi **public** được vẽ với **màu đỏ** bên phải.
  - + Các thuộc tính và phương thức có được do kế thừa được vẽ bằng nét đứt không liên tục.
  - + Các thuộc tính và phương thức của chính bản thân lớp được vẽ bằng nét liền liên tục.

## 6. CÂY KẾ THỪA CHI TIẾT



## 6. CÂY KẾ THỪA CHI TIẾT



## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Ví dụ dẫn nhập 01: Hãy cho biết trong chương trình dưới đây câu lệnh nào đúng câu lệnh nào sai:

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main( )
20.{
21.|    A a;
22.|    B b;
23.|    a = b;
24.|    b = a;
25.}
```

## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Ví dụ dẫn nhập 02: Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai:

```
1. class A
2. {
3. |    ...
4. } ;
```

```
5. class B: public A
6. {
7. |    ...
8. } ;
```

```
9. void main( )
10. {
11. |    A *a;
12. |    B *b;
13. |    A x;
14. |    B y;
15. |    a = &x;
16. |    b = &y;
17. |    a = &y;
18. |    b = &x;
19. }
```

## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Toán tử gán trong kế thừa được thực hiện theo nguyên tắc: **trường hợp đặt biệt có thể được gán cho trường hợp tổng quát, và trường hợp tổng quát thì không thể gán cho trường hợp đặt biệt được.**
- Qui tắc trên áp dụng cho tất cả các ngôn ngữ hỗ trợ lập trình hướng đối tượng như C++, Java, VB.NET, C#, Python,...

## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Áp dụng qui tắc trên cho ngôn ngữ lập trình hướng đối tượng C++ ta có thể nói như sau: **một đối tượng thuộc lớp dẫn xuất có thể được gán cho một đối tượng thuộc lớp cơ sở**. Điều ngược lại là sai, nghĩa là một đối tượng thuộc lớp cơ sở không được quyền gán cho một đối tượng thuộc lớp dẫn xuất.



## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Ví dụ dẫn nhập 01: Hãy cho biết trong chương trình dưới đây câu lệnh nào đúng câu lệnh nào sai:

```
11.class A
12.{
13.|    ...
14.};
15.class B:public A
16.{
17.|    ...
18.};
19.void main( )
20.{
21.|    A a;
22.|    B b;
23.|    a = b;
24.|    b = a;
25.}
```

## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Mở rộng qui tắc trên cho con trỏ đối tượng ta có thể nói như sau: **một con trỏ đối tượng thuộc lớp cơ sở có thể giữ địa chỉ của một đối tượng thuộc lớp dẫn xuất. Ngược lại, một con trỏ đối tượng thuộc lớp dẫn xuất không thể giữ địa chỉ của một đối tượng thuộc lớp cơ sở.**

## 7. TOÁN TỬ GÁN TRONG KẾ THỪA

- Hãy cho biết đoạn chương trình dưới đây câu lệnh nào đúng, câu lệnh nào sai:

```
11. class A
12. {
13. };
14. class B:public A
15. {
16. };
17. void main( )
18. {
19.     A *a;
20.     B *b;
21.     A x;
22.     B y;
23.     a = &x;
24.     b = &y;
25.     a = &y;
26.     b = &x;
27. }
```